# UAV-Vehicle Interaction Simulation Platform and UAV Riding Strategy Verification

**Yao He**

**Network Communications and Economics Lab (NCEL)**

School of Science and Engineering

The Chinese University of Hong Kong, Shenzhen (CUHKSZ)

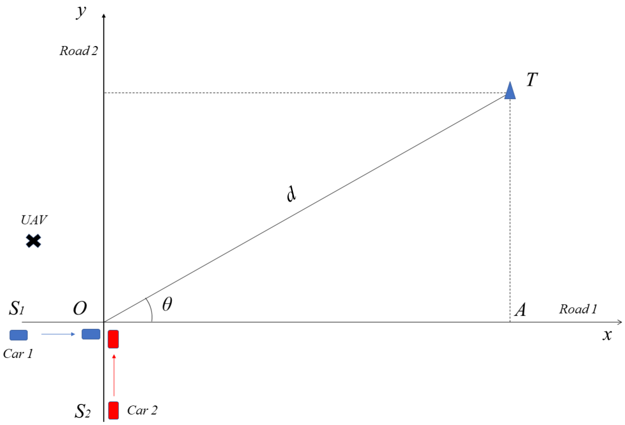
# Motivation

- Build a UAV-Vehicle interaction simulation platform
- Verify the performance of UAV riding strategy
- Potential extension for UAV related researches

# Outline

- UAV riding strategy
- Platform and my work
- Verification results

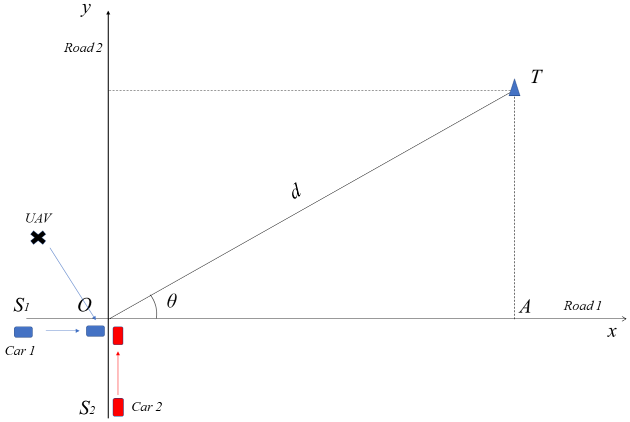
# Simulation Scenario




UAV-Vehicle cooperation map

- Simulation Stages:
  - Speed recognition: Car starts to drive from S1/S2, and stops at O. This stage, the UAV will collect the speed of the cars and choose one car to land on.

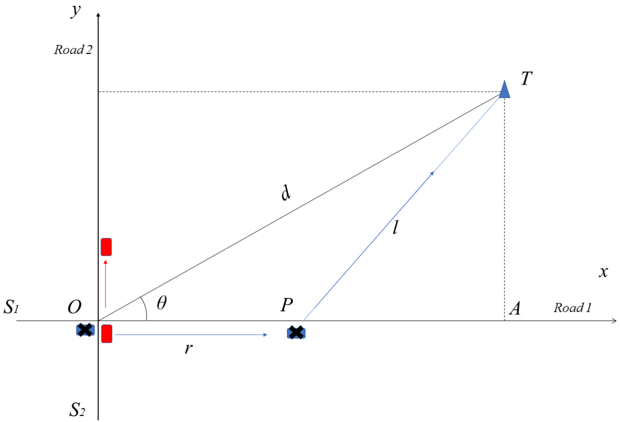
# Simulation Scenario




UAV-Vehicle cooperation map

- Simulation Stages:
  - Speed recognition: Car starts to drive from S1/S2, and stops at O.
  - Landing: When cars stops at O, the UAV flies and lands onto the chosen car.

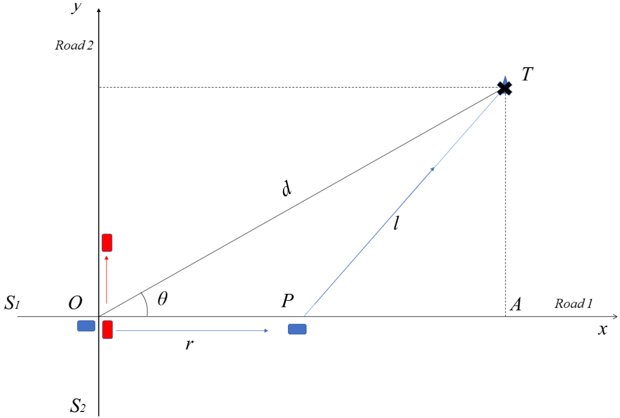
# Simulation Scenario




UAV-Vehicle cooperation map

- Simulation Stages:
  - Car carries UAV: After UAV lands onto the car, the car drives along OA, carrying the UAV. Time starts to count.
  - Take off: At some point P between OA, the UAV takes off and flies to T along PT. The energy consumption starts to count.

# Simulation Scenario

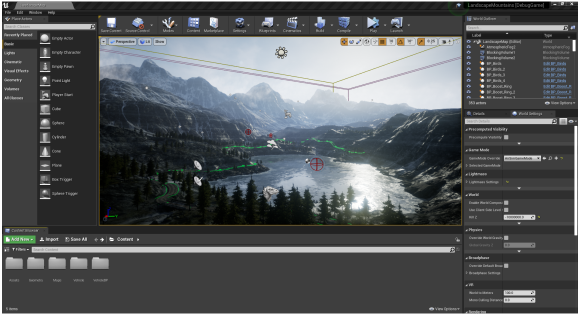



UAV-Vehicle cooperation map

- Simulation Stages:
  - Take off: At some point P between OA, the UAV takes off and flies to T along PT. The energy consumption starts to count.
  - Termination: UAV reaches T. Data collection stops, and experiment completes.

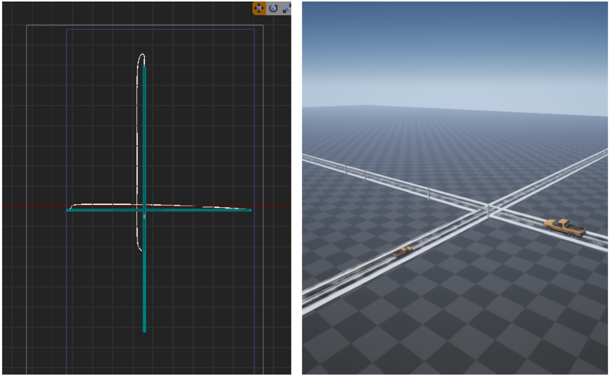
# Platform: Unreal Engine 4 + AirSim

- Unreal Engine 4 (UE4)
  - Main simulation engine
  - Place where the simulation is designed and tested
  - UE4 project:
    - Static Meshes: Static elements that provide the scenescape.
    - Actors: Potentially contains a variety of Static Meshes, and perform specific tasks according to programs.




Unreal Engine 4 overview

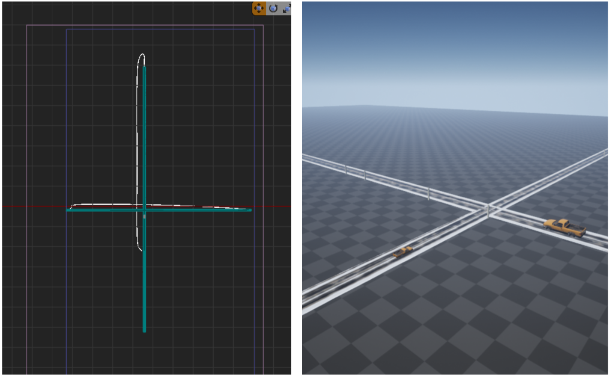
# UE4 Simulation Environment



UE4 simulation overview.

- Static Meshes
  - 2 roads.
  - Objects called path. They inform AI cars to travel on specific routes.
  - Objects called marks which help construct a coordinate system in the testing scripts.

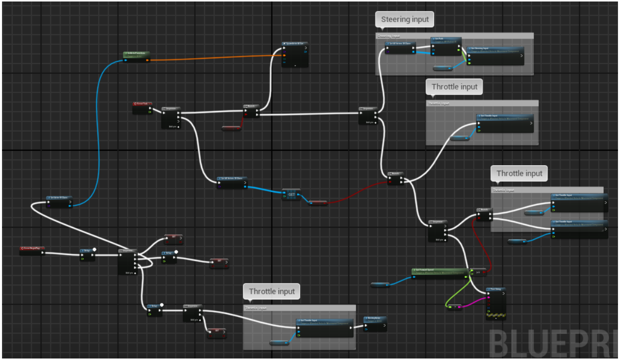
# UE4 Simulation Environment



UE4 simulation overview.

- Actors
  - AI cars that automatically drive along path.

# AI Cars




Scripts for AI cars overview.

- AI cars function
  - A function called getPath(), that gives the path information to the cars.
  - A control function that stabilize the velocity of the vehicles.
  - Keyboard instructions that control the driving of cars if needed.

# Platform: Unreal Engine 4 + AirSim

Unreal Engine 4 + AirSim

- AirSim
  - A plugin of UE4.
  - Provides UAV models.
  - Exposes APIs(Application Programming Interface) so we can interact with the AirSim vehicles and collect real-time data in the simulation programmatically.

UAV model from AirSim.

# AirSim APIs

- World APIs: Environmental setting and collecting data
  - Enabling wind and fogs.
  - Listing objects.
  - Collecting GPS data.
  - Images.
  - ...
- Drone APIs: Control the movement of UAVs.
  - Landing and taking off.
  - Flying to target position.
  - Rotation.
  - ...
- Vehicle APIs: Control the movement of AirSim cars (not the AI cars).

# Energy Consumption

The power of the UAV model used in AirSim is

$$P = Tv_h$$

where $T$ is the thrust and $v_h$ the air velocity. $T$ and $v_h$ can be obtained using AirSim APIs. Energy consumption is simply an integration of power over time

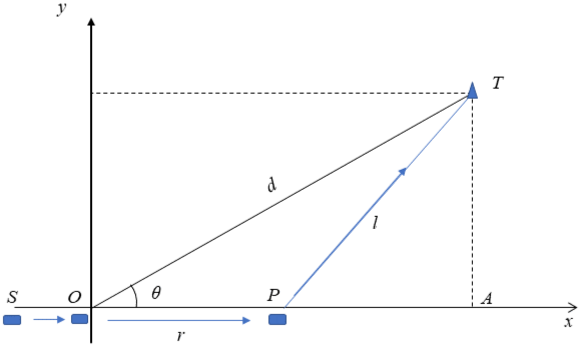
$$E(r) = \int P(t)dt$$

# NCEL_lib.py

Integrate the AirSim APIs with extra functions to a library made for our lab.

- Objective:
  - Bug free.
  - Stabilize experiments.
  - Easy to understand and use.
  - Integrate AirSim with our new functions.
  - Flexible to extend.
- Features:
  - Improvements on the AirSim APIs.
  - A coordinate system and transformation system.
  - A simple algorithm that controls the landing of UAVs.
  - A function that computes the energy of UAVs.
  - A new API that computes vehicles' speeds.

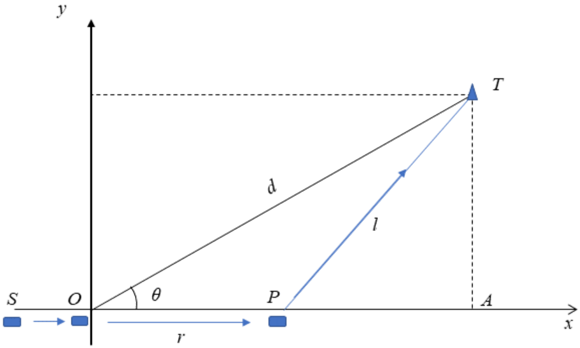
# Analysis of the scenario




UAV-Vehicle cooperation map

- Time Consumption $T(r) = \frac{r}{V_{car}} + \frac{l}{V_{UAV}}$, where $V_{car}$ and $V_{UAV}$ denote the speed of car and the UAV, respectively.
- Energy consumption $E(r) = \int_{\frac{r}{V_{car}}}^{T(r)} P(t)dt$, where $P(t)$ is the power of UAV.

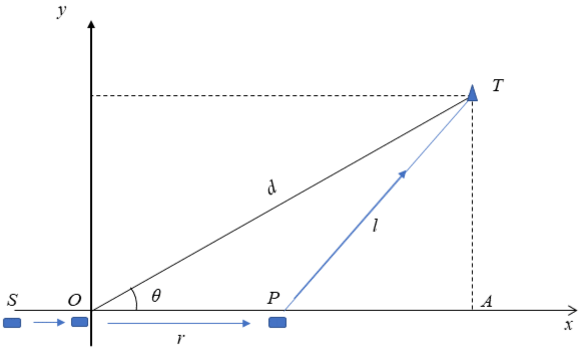
# Analysis of the scenario




UAV-Vehicle cooperation map

- Cost $C(r) = \omega \times E_n(r) + (1-\omega) \times T(r)$, where $\omega$ is the trade off parameter and $0 \leq \omega \leq 1$. $E_n(r)$ is the normalized energy consumption.
- $E_n(r) = \frac{E(r)}{\alpha}$, where $\alpha$ indicates the unit energy consumption in each second, which we should determine in the experiment.

# Analysis of the scenario

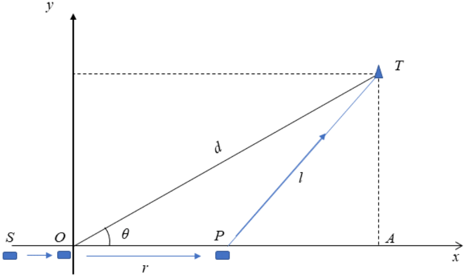



UAV-Vehicle cooperation map

- $E_n(r) = \frac{E(r)}{\alpha}$, where $\alpha$ indicates the unit energy consumption in each second, which we should determine in the experiment.
- If the $\alpha$ is constant, then $E_n(r) = \frac{l}{V_{UAV}}$

# Problem

$$
\begin{array}{rll}
\textbf{minimize} & \omega \times E_n(r) + (1-\omega) \times T(r) & \\
\textbf{subject to} & \text{(i)} \quad T(r) = \frac{r}{V_{car}} + \frac{l}{V_{UAV}} & \\
 & \text{(ii)} \quad E_n(r) = \frac{E(r)}{\alpha} = \frac{l}{V_{UAV}} & \text{for constant } \alpha \\
 & \text{(iii)} \quad l = \sqrt{d^2 - 2dr\cos\theta + r^2} & \\
\textbf{variables} & r &
\end{array}
$$

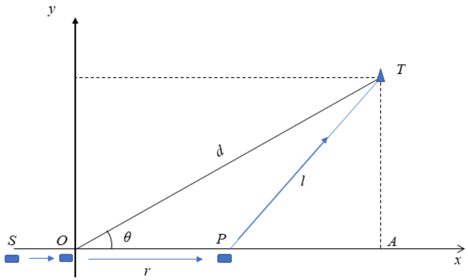



$$
\begin{array}{rl}
\textbf{minimize} & \omega \times E_n(r) + (1-\omega) \times T(r) \\
\textbf{subject to} & \text{(i)} \quad T(r) = \frac{r}{V_{car}} + \frac{l}{V_{UAV}} \\
& \text{(ii)} \quad E_n(r) = \frac{E(r)}{\alpha} = \frac{l}{V_{UAV}} \qquad \text{for constant } \alpha \\
& \text{(ii)} \quad l = \sqrt{d^2 - 2dr\cos\theta + r^2} \\
\textbf{variables} & r
\end{array}
$$

The optimal solution is given by

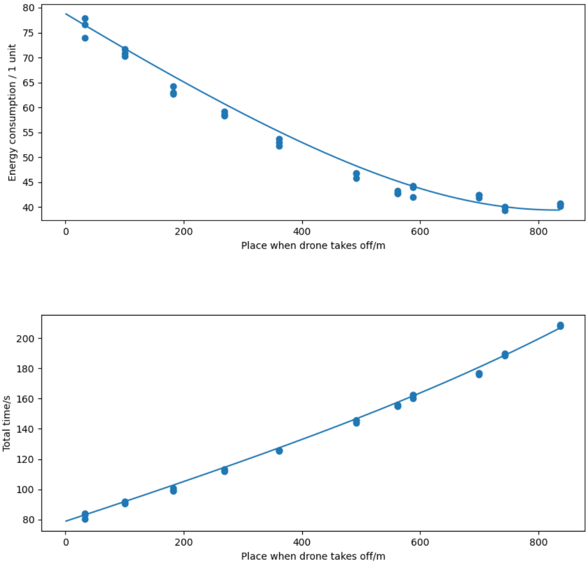
$$
r^* = d\cos\theta - d\sin\theta \times \frac{(1-\omega)V_{UAV}}{\sqrt{V_{car}^2 - (1-\omega^2)V_{UAV}^2}}
$$

# Parameter Setting

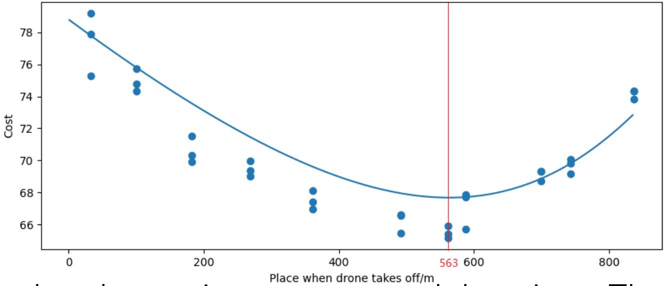



- $\theta = 30^\circ$
- $d = 960m$
- Measured $V_{UAV} = 12.2m/s$
- Measured $V_{car} = 5m/s$
- Measured unit energy consumption $\alpha = 63$
- Trade off parameter $\omega = 0.8$

# Measured data in the simulator.



# Cost




- For each $r$, the experiments are repeated three times. The standard deviations of recorded time, normalized energy and cost are 0.635, 0.583, and 0.588, respectively.
- The theoretical optimal value of take off position $r^*$ is around $563m$.
- The experimental optimal value of take off position $r^*$ is around $563m$.

# Conclusion

- Build a UAV-Vehicle interaction simulation platform
- Verify the performance of UAV riding strategy
- Potential extension for UAV related researches

# Q & A

# References


- AirSim documentations: https://www.zhihu.com/column/multiUAV
- AirSim repository: https://microsoft.github.io/AirSim/
- Shital, S., Debadeepta, D., Chris, L., and Ashish, K., AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles, 2017, Field and Service Robotics, arXiv:1705.05065, https://arxiv.org/abs/1705.05065
- Unreal Engine 4 documentations: https://docs.unrealengine.com/en-US/index.html
- Nicola, Z., Multirotor Aircraft Dynamics, Simulation and Control, 2016, IEEE Computer Society, https://www.researchgate.net/publication/307513305